19 June 2024

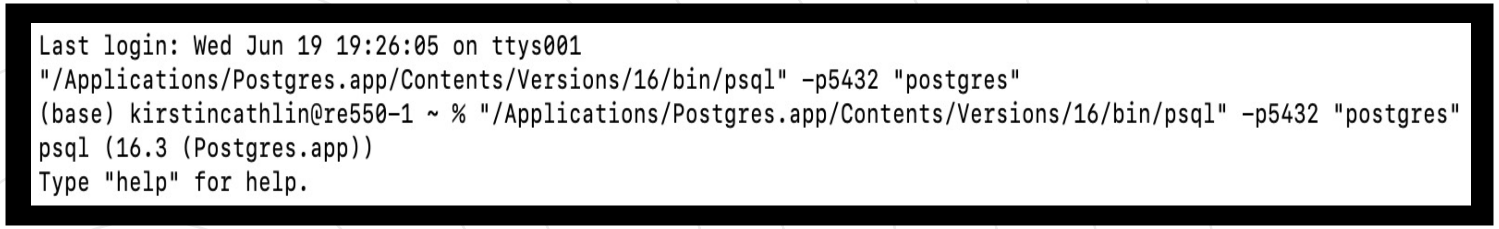
Development Phase: Build a Data Mart in SQL

KIRSTIN MOFFATT (92128298)
DLBDSPBDM01

SETUP & RUN

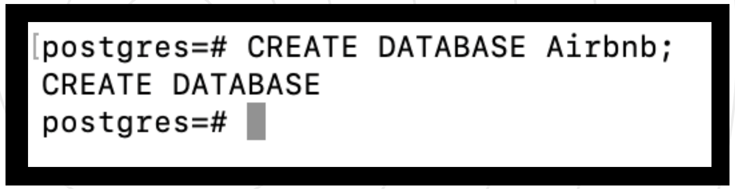
Setup PostgreSQL

01. DOWNLOAD PostgreSQL VIA Homebrew



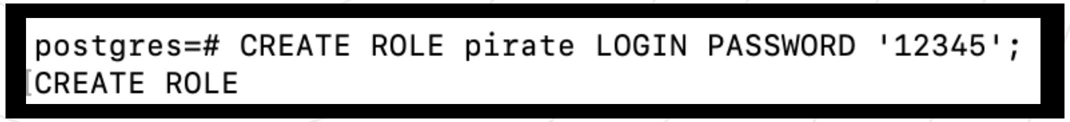
```
Last login: Wed Jun 19 19:26:05 on ttys001
"/Applications/Postgres.app/Contents/Versions/16/bin/psql" -p5432 "postgres"
(base) kirstincathlin@re550-1 ~ % "/Applications/Postgres.app/Contents/Versions/16/bin/psql" -p5432 "postgres"
psql (16.3 (Postgres.app))
Type "help" for help.
```

02. CREATE DATABASE



```
postgres=# CREATE DATABASE Airbnb;
CREATE DATABASE
postgres=#
```

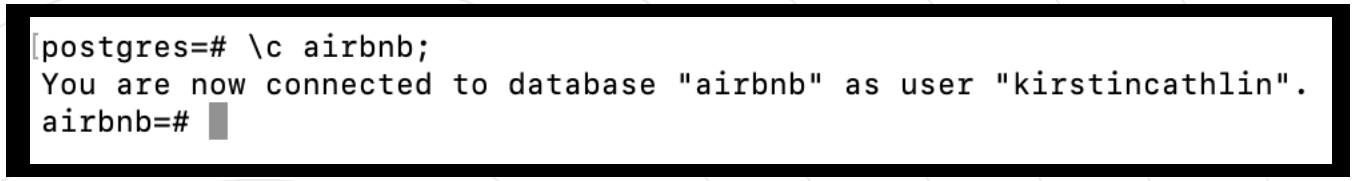
03. CREATE ROLE



```
postgres=# CREATE ROLE pirate LOGIN PASSWORD '12345';
CREATE ROLE
```

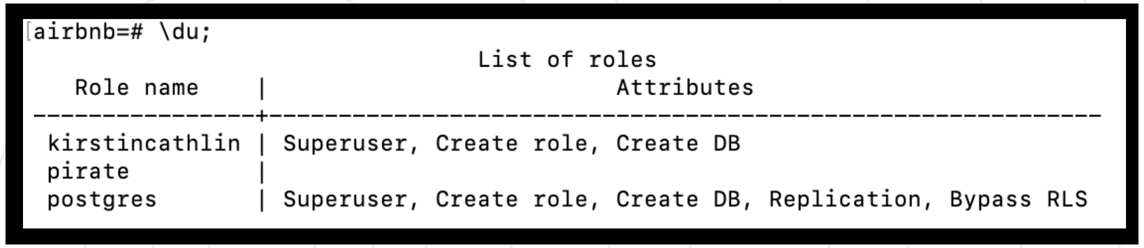
Setup PostgreSQL

04. CONNECT TO DATABASE



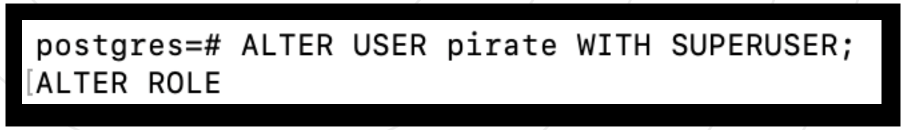
```
postgres=# \c airbnb;
You are now connected to database "airbnb" as user "kirstincathlin".
airbnb=#
```

05. VIEW EXISTING USER PERMISSIONS



```
airbnb=# \du;
                                   List of roles
   Role name    |                         Attributes
----------------+------------------------------------------------------------
 kirstincathlin | Superuser, Create role, Create DB
 pirate         |
 postgres       | Superuser, Create role, Create DB, Replication, Bypass RLS
```

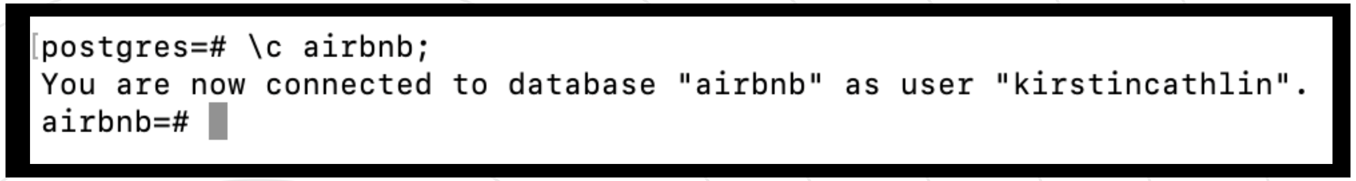
06. GRANT pirate user SUPERUSER PRIVILEGES



```
postgres=# ALTER USER pirate WITH SUPERUSER;
ALTER ROLE
```

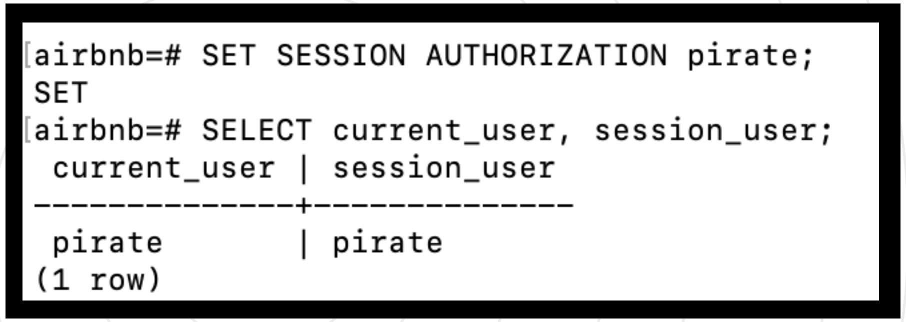
Setup PostgreSQL

07. CHECK USER PERMISSIONS AGAIN



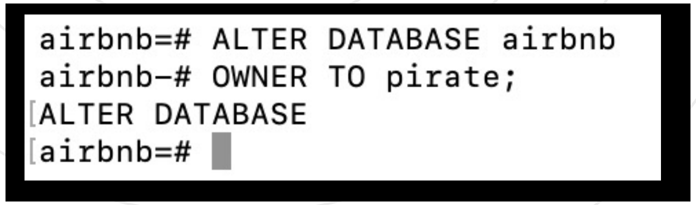
```
postgres=# \c airbnb;
You are now connected to database "airbnb" as user "kirstincathlin".
airbnb=#
```

05. CHANGE SESSION_USER & CHECK AUTHORIZATIONS

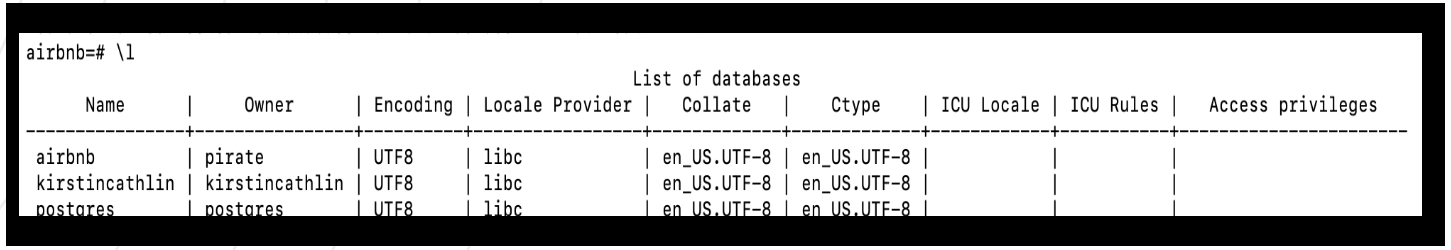


```
airbnb=# SET SESSION AUTHORIZATION pirate;
SET
airbnb=# SELECT current_user, session_user;
 current_user | session_user
--------------+--------------
 pirate       | pirate
(1 row)
```

06. CHANGE AIRBNB DATABASE OWNER TO PIRATE



```
airbnb=# ALTER DATABASE airbnb
airbnb-# OWNER TO pirate;
ALTER DATABASE
airbnb=#
```

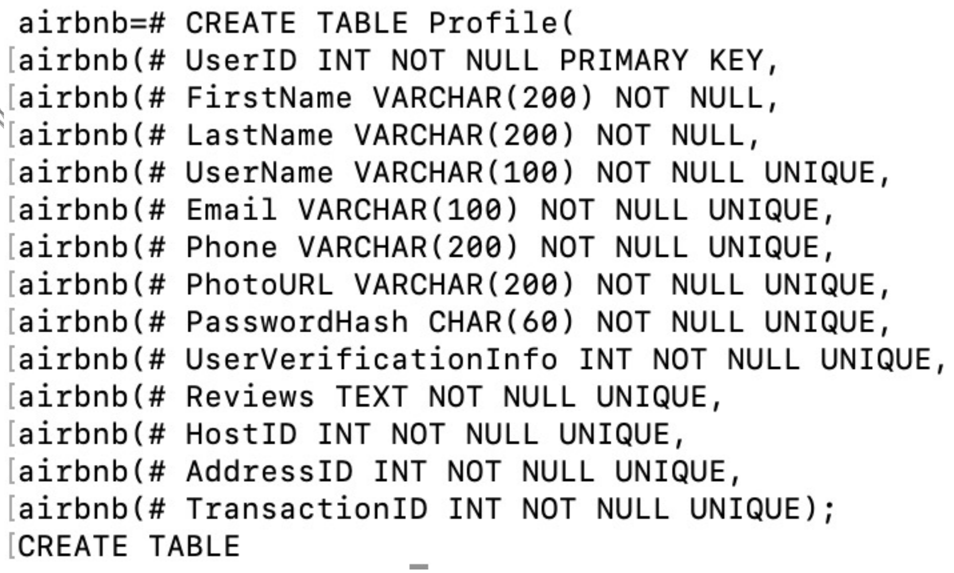


```
airbnb=# \l
                                                        List of databases
      Name      |     Owner      | Encoding | Locale Provider |   Collate   |    Ctype    | ICU Locale | ICU Rules |   Access privileges
----------------+----------------+----------+-----------------+-------------+-------------+------------+-----------+-----------------------
 airbnb         | pirate         | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |            |           |
 kirstincathlin | kirstincathlin | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |            |           |
 postgres       | postgres       | UTF8     | libc            | en_US.UTF-8 | en_US.UTF-8 |            |           |
```

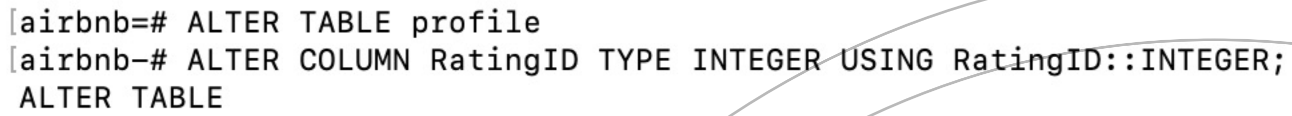
TABLE CREATION

Refer to corresponding data dictionary for definitions

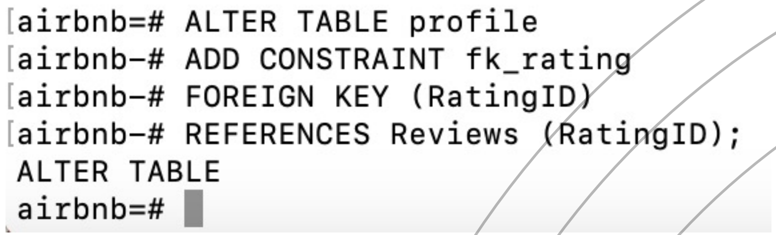
Profile



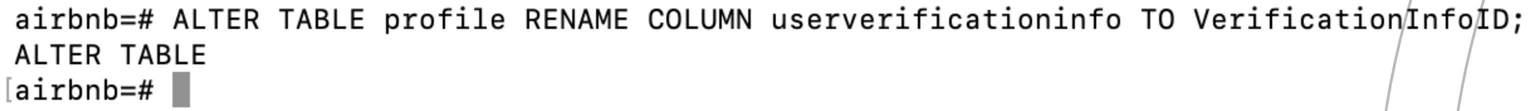
```
airbnb=# CREATE TABLE Profile(
[airbnb(# UserID INT NOT NULL PRIMARY KEY,
[airbnb(# FirstName VARCHAR(200) NOT NULL,
[airbnb(# LastName VARCHAR(200) NOT NULL,
[airbnb(# UserName VARCHAR(100) NOT NULL UNIQUE,
[airbnb(# Email VARCHAR(100) NOT NULL UNIQUE,
[airbnb(# Phone VARCHAR(200) NOT NULL UNIQUE,
[airbnb(# PhotoURL VARCHAR(200) NOT NULL UNIQUE,
[airbnb(# PasswordHash CHAR(60) NOT NULL UNIQUE,
[airbnb(# UserVerificationInfo INT NOT NULL UNIQUE,
[airbnb(# Reviews TEXT NOT NULL UNIQUE,
[airbnb(# HostID INT NOT NULL UNIQUE,
[airbnb(# AddressID INT NOT NULL UNIQUE,
[airbnb(# TransactionID INT NOT NULL UNIQUE);
[CREATE TABLE
```



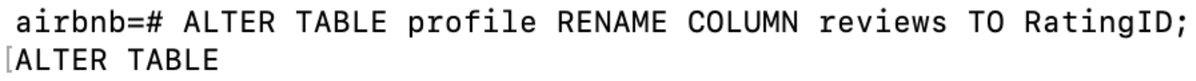
```
[airbnb=# ALTER TABLE profile
[airbnb-# ALTER COLUMN RatingID TYPE INTEGER USING RatingID::INTEGER;
ALTER TABLE
```



```
[airbnb=# ALTER TABLE profile
[airbnb-# ADD CONSTRAINT fk_rating
[airbnb-# FOREIGN KEY (RatingID)
[airbnb-# REFERENCES Reviews (RatingID);
ALTER TABLE
airbnb=#
```

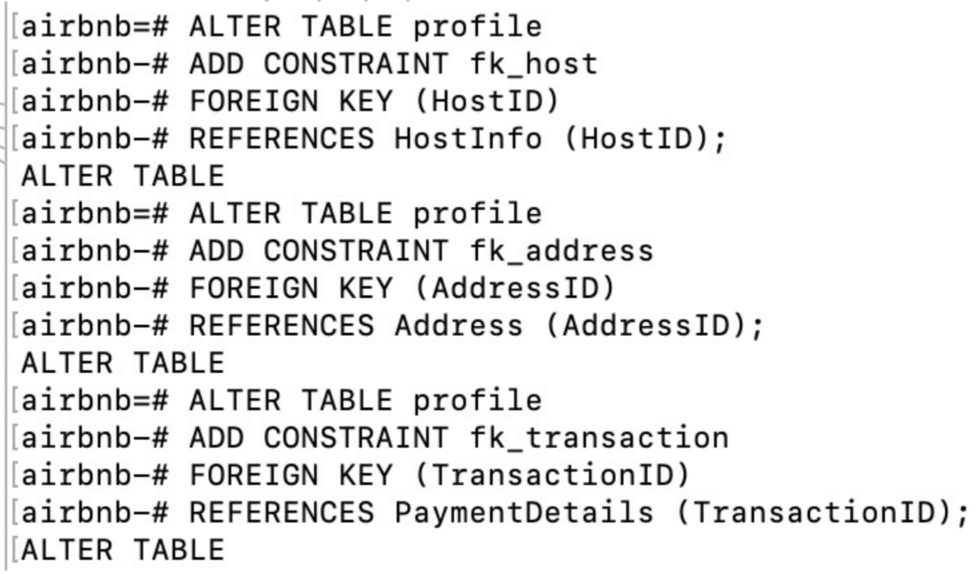


```
airbnb=# ALTER TABLE profile RENAME COLUMN userverificationinfo TO VerificationInfoID;
ALTER TABLE
[airbnb=#
```

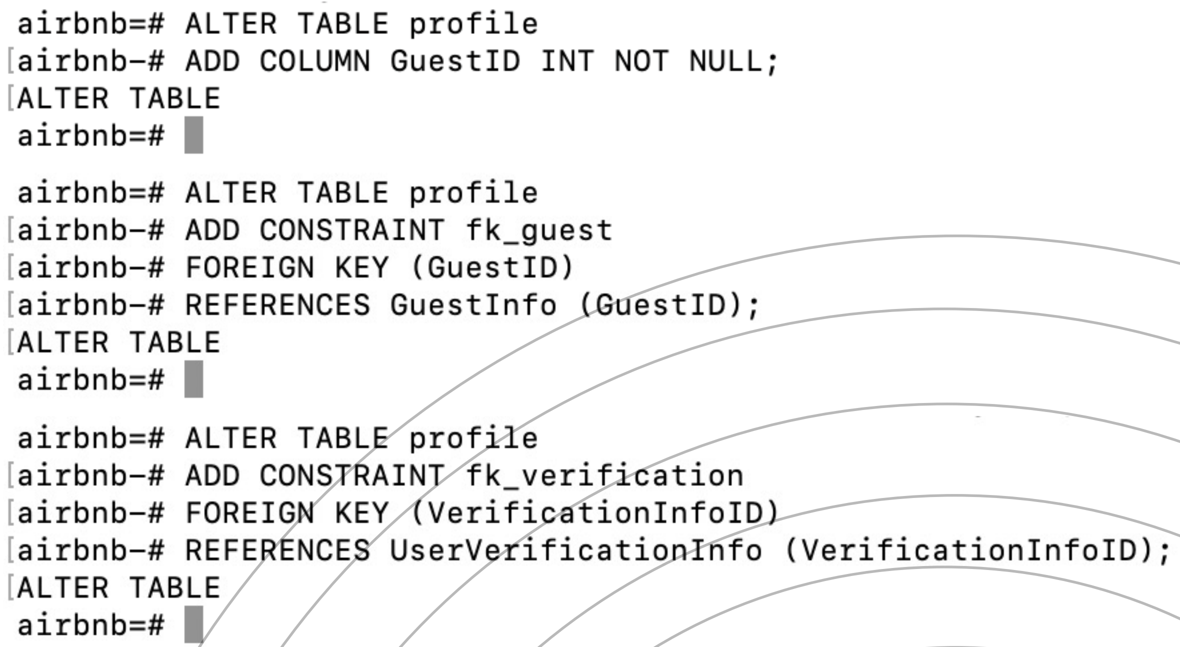


```
airbnb=# ALTER TABLE profile RENAME COLUMN reviews TO RatingID;
[ALTER TABLE
```

Profile

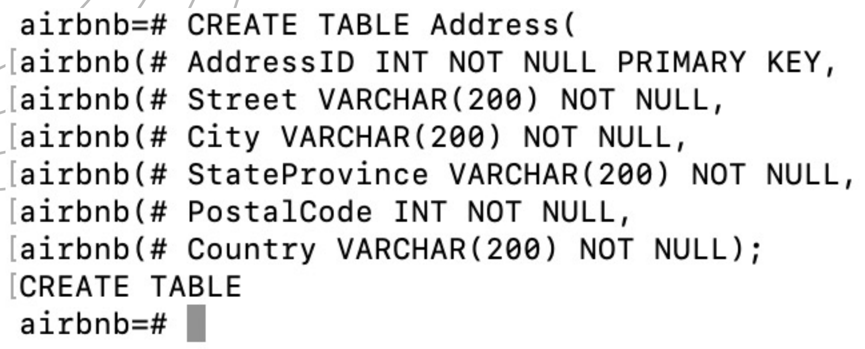


```
[airbnb=# ALTER TABLE profile
[airbnb-# ADD CONSTRAINT fk_host
[airbnb-# FOREIGN KEY (HostID)
[airbnb-# REFERENCES HostInfo (HostID);
 ALTER TABLE
[airbnb=# ALTER TABLE profile
[airbnb-# ADD CONSTRAINT fk_address
[airbnb-# FOREIGN KEY (AddressID)
[airbnb-# REFERENCES Address (AddressID);
 ALTER TABLE
[airbnb=# ALTER TABLE profile
[airbnb-# ADD CONSTRAINT fk_transaction
[airbnb-# FOREIGN KEY (TransactionID)
[airbnb-# REFERENCES PaymentDetails (TransactionID);
[ALTER TABLE
```



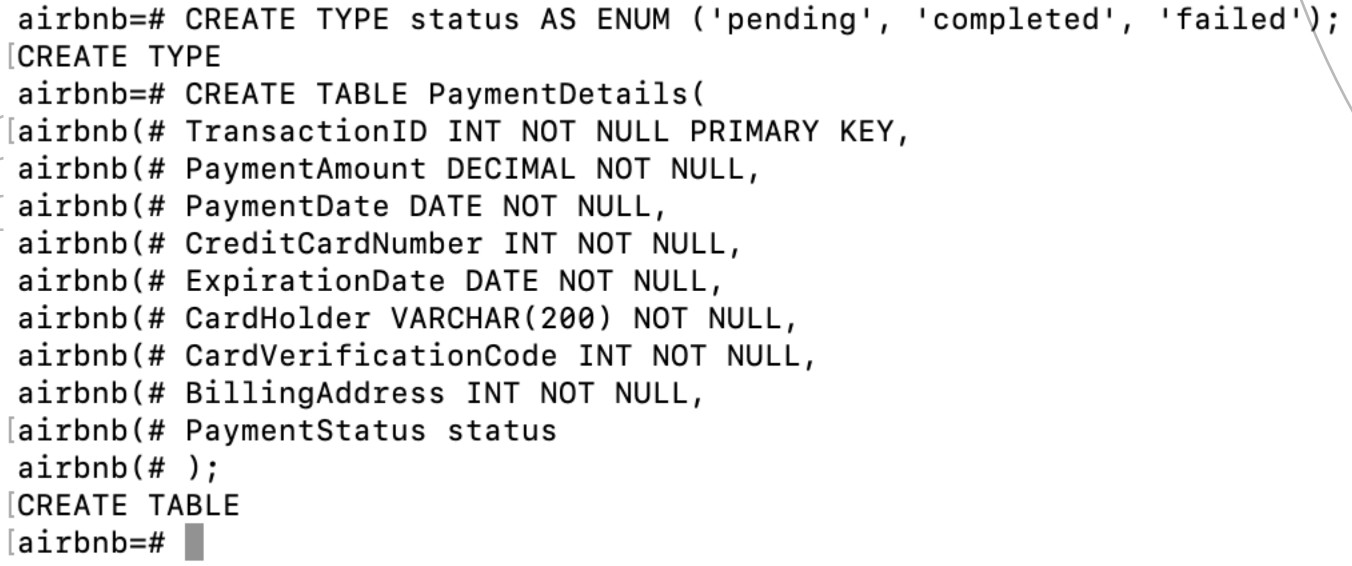
```
 airbnb=# ALTER TABLE profile
[airbnb-# ADD COLUMN GuestID INT NOT NULL;
[ALTER TABLE
 airbnb=#

 airbnb=# ALTER TABLE profile
[airbnb-# ADD CONSTRAINT fk_guest
[airbnb-# FOREIGN KEY (GuestID)
[airbnb-# REFERENCES GuestInfo (GuestID);
[ALTER TABLE
 airbnb=#

 airbnb=# ALTER TABLE profile
[airbnb-# ADD CONSTRAINT fk_verification
[airbnb-# FOREIGN KEY (VerificationInfoID)
[airbnb-# REFERENCES UserVerificationInfo (VerificationInfoID);
[ALTER TABLE
 airbnb=#
```

Address

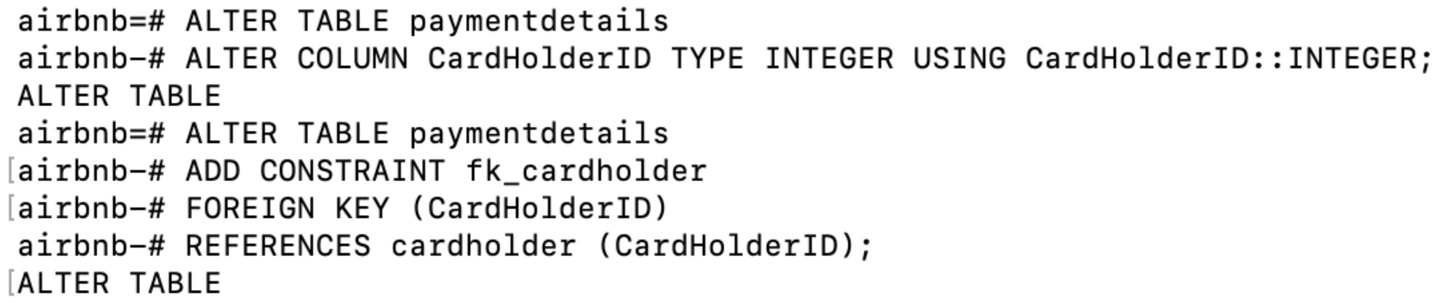


```
 airbnb=# CREATE TABLE Address(
[airbnb(# AddressID INT NOT NULL PRIMARY KEY,
[airbnb(# Street VARCHAR(200) NOT NULL,
[airbnb(# City VARCHAR(200) NOT NULL,
[airbnb(# StateProvince VARCHAR(200) NOT NULL,
[airbnb(# PostalCode INT NOT NULL,
[airbnb(# Country VARCHAR(200) NOT NULL);
[CREATE TABLE
 airbnb=# 
```

Payment Details

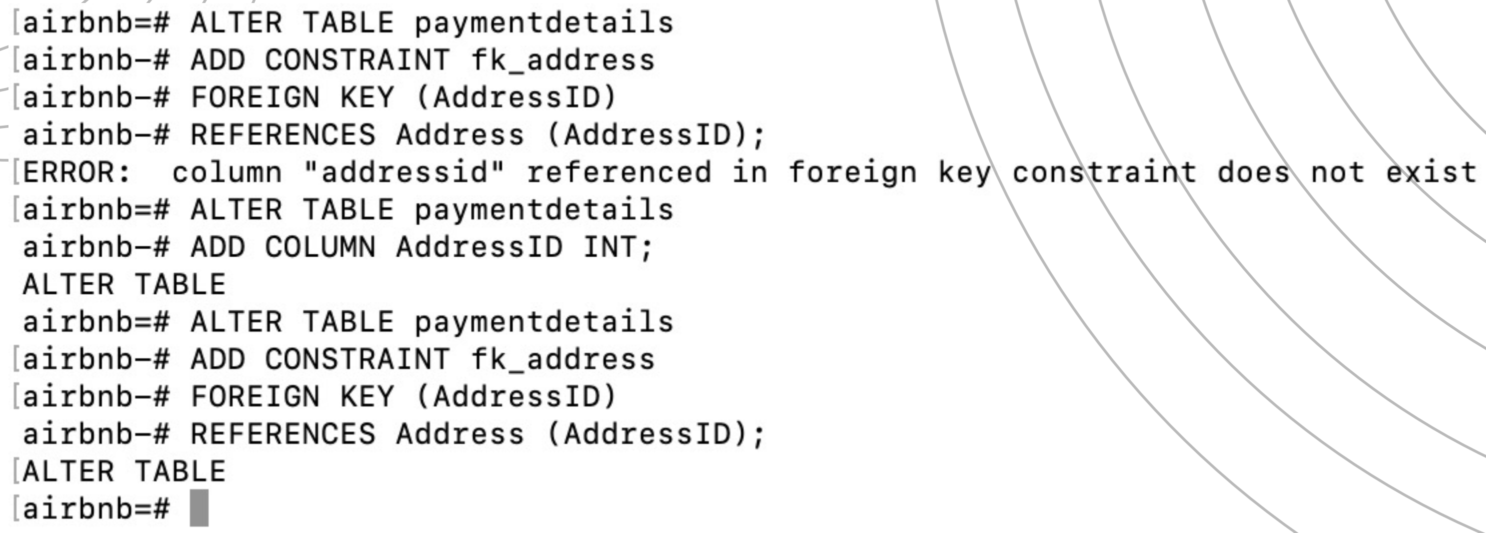


```
airbnb=# CREATE TYPE status AS ENUM ('pending', 'completed', 'failed');
CREATE TYPE
airbnb=# CREATE TABLE PaymentDetails(
airbnb(# TransactionID INT NOT NULL PRIMARY KEY,
airbnb(# PaymentAmount DECIMAL NOT NULL,
airbnb(# PaymentDate DATE NOT NULL,
airbnb(# CreditCardNumber INT NOT NULL,
airbnb(# ExpirationDate DATE NOT NULL,
airbnb(# CardHolder VARCHAR(200) NOT NULL,
airbnb(# CardVerificationCode INT NOT NULL,
airbnb(# BillingAddress INT NOT NULL,
airbnb(# PaymentStatus status
airbnb(# );
CREATE TABLE
airbnb=#
```

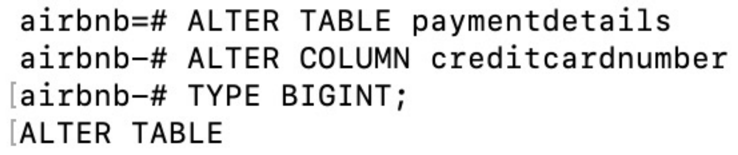


```
airbnb=# ALTER TABLE paymentdetails
airbnb-# ALTER COLUMN CardHolderID TYPE INTEGER USING CardHolderID::INTEGER;
ALTER TABLE
airbnb=# ALTER TABLE paymentdetails
airbnb-# ADD CONSTRAINT fk_cardholder
airbnb-# FOREIGN KEY (CardHolderID)
airbnb-# REFERENCES cardholder (CardHolderID);
ALTER TABLE
```

Payment Details

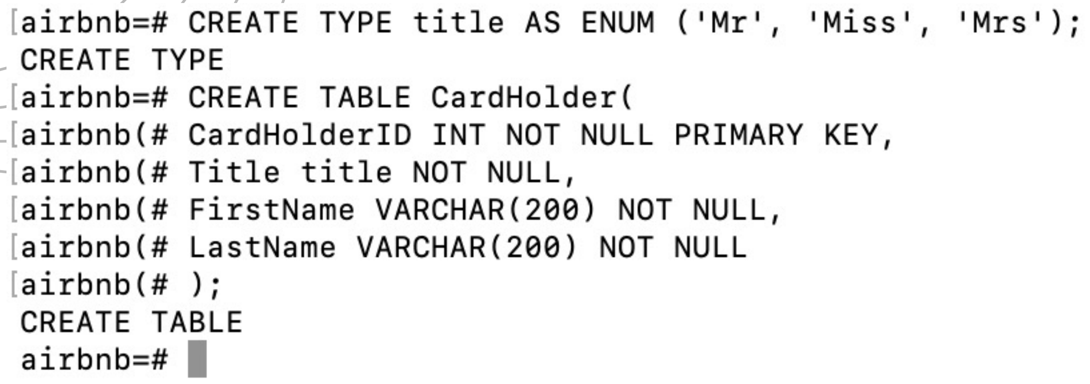


```
[airbnb=# ALTER TABLE paymentdetails
[airbnb-# ADD CONSTRAINT fk_address
[airbnb-# FOREIGN KEY (AddressID)
 airbnb-# REFERENCES Address (AddressID);
[ERROR:  column "addressid" referenced in foreign key constraint does not exist
[airbnb=# ALTER TABLE paymentdetails
 airbnb-# ADD COLUMN AddressID INT;
 ALTER TABLE
 airbnb=# ALTER TABLE paymentdetails
[airbnb-# ADD CONSTRAINT fk_address
[airbnb-# FOREIGN KEY (AddressID)
 airbnb-# REFERENCES Address (AddressID);
[ALTER TABLE
[airbnb=#
```



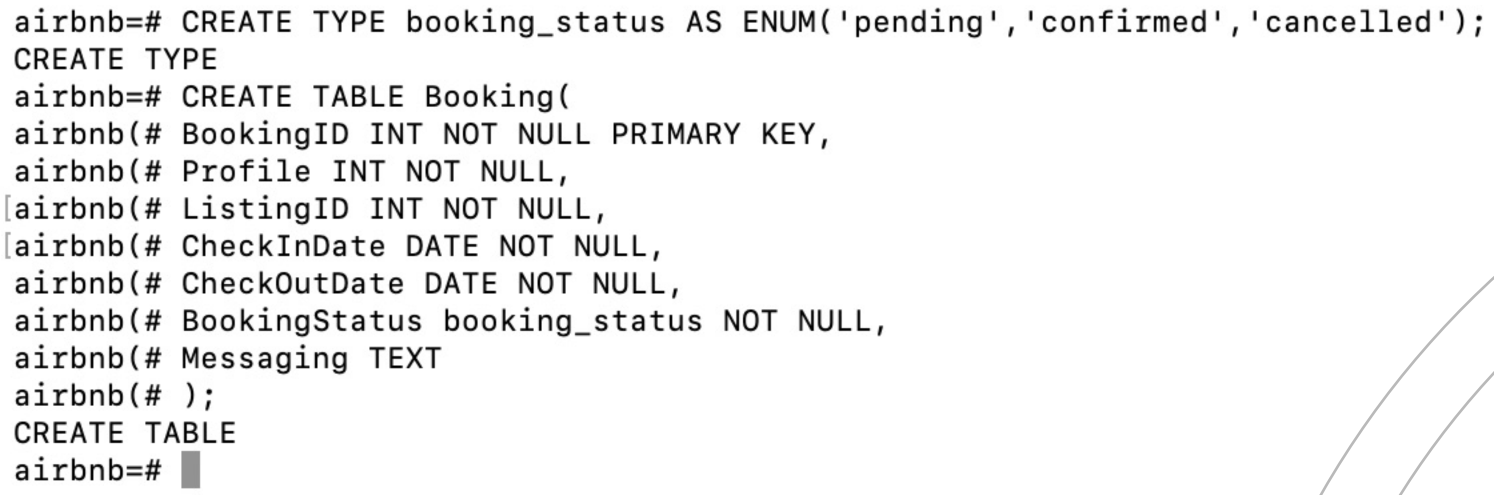
```
 airbnb=# ALTER TABLE paymentdetails
 airbnb-# ALTER COLUMN creditcardnumber
[airbnb-# TYPE BIGINT;
[ALTER TABLE
```

Card Holder

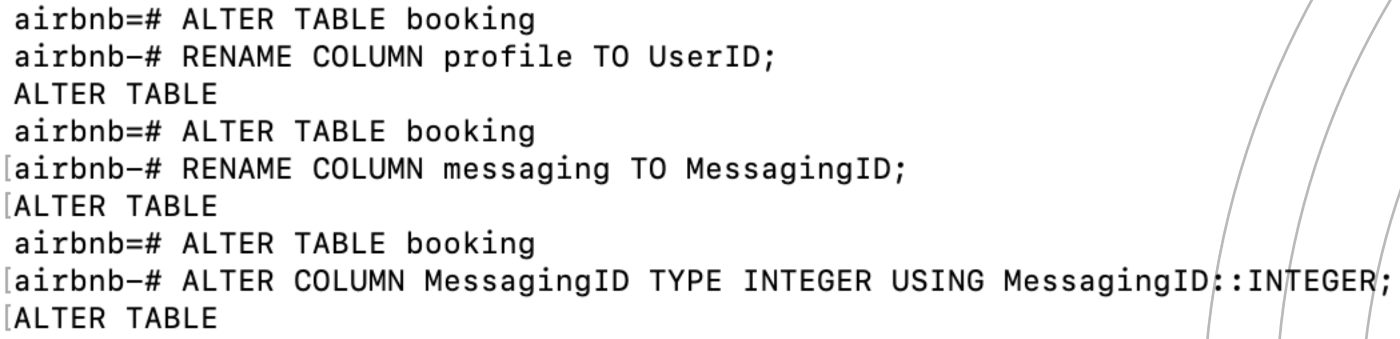


```
[airbnb=# CREATE TYPE title AS ENUM ('Mr', 'Miss', 'Mrs');
CREATE TYPE
[airbnb=# CREATE TABLE CardHolder(
[airbnb(# CardHolderID INT NOT NULL PRIMARY KEY,
[airbnb(# Title title NOT NULL,
[airbnb(# FirstName VARCHAR(200) NOT NULL,
[airbnb(# LastName VARCHAR(200) NOT NULL
[airbnb(# );
CREATE TABLE
airbnb=#
```

Booking

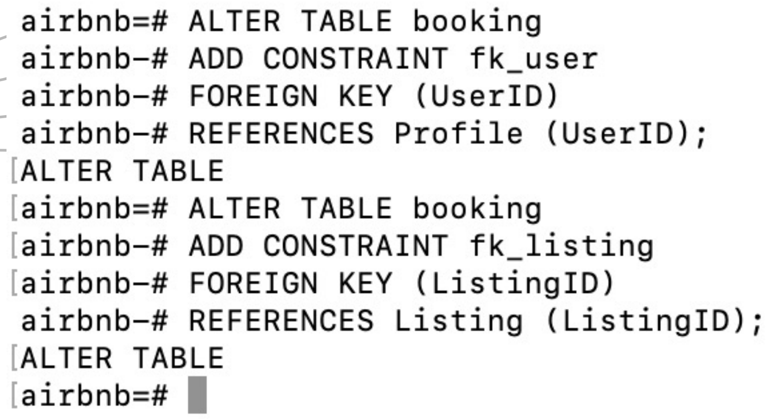


```
airbnb=# CREATE TYPE booking_status AS ENUM('pending','confirmed','cancelled');
CREATE TYPE
airbnb=# CREATE TABLE Booking(
airbnb(# BookingID INT NOT NULL PRIMARY KEY,
airbnb(# Profile INT NOT NULL,
airbnb(# ListingID INT NOT NULL,
airbnb(# CheckInDate DATE NOT NULL,
airbnb(# CheckOutDate DATE NOT NULL,
airbnb(# BookingStatus booking_status NOT NULL,
airbnb(# Messaging TEXT
airbnb(# );
CREATE TABLE
airbnb=# 
```



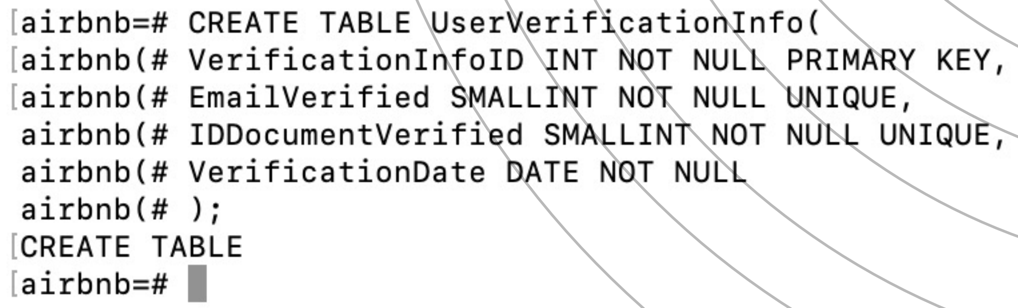
```
airbnb=# ALTER TABLE booking
airbnb-# RENAME COLUMN profile TO UserID;
ALTER TABLE
airbnb=# ALTER TABLE booking
airbnb-# RENAME COLUMN messaging TO MessagingID;
ALTER TABLE
airbnb=# ALTER TABLE booking
airbnb-# ALTER COLUMN MessagingID TYPE INTEGER USING MessagingID::INTEGER;
ALTER TABLE
```

Booking



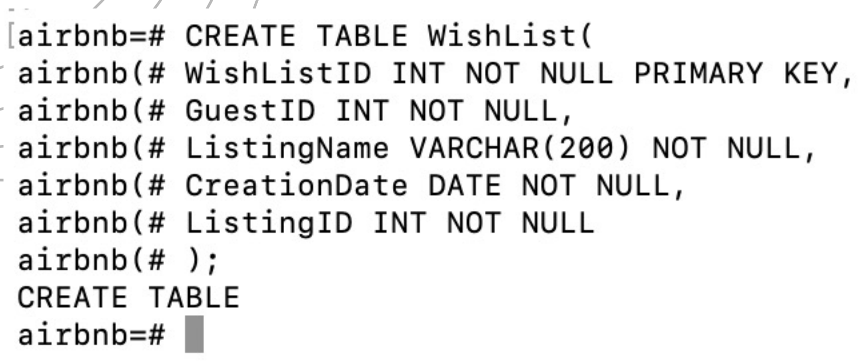
```
 airbnb=# ALTER TABLE booking
 airbnb-# ADD CONSTRAINT fk_user
 airbnb-# FOREIGN KEY (UserID)
 airbnb-# REFERENCES Profile (UserID);
[ALTER TABLE
[airbnb=# ALTER TABLE booking
[airbnb-# ADD CONSTRAINT fk_listing
[airbnb-# FOREIGN KEY (ListingID)
 airbnb-# REFERENCES Listing (ListingID);
[ALTER TABLE
[airbnb=# 
```

User Verification Info

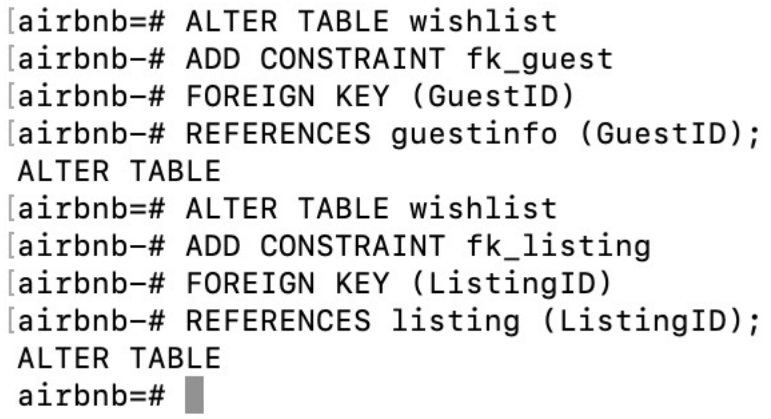


```
[airbnb=# CREATE TABLE UserVerificationInfo(
[airbnb(# VerificationInfoID INT NOT NULL PRIMARY KEY,
[airbnb(# EmailVerified SMALLINT NOT NULL UNIQUE,
 airbnb(# IDDocumentVerified SMALLINT NOT NULL UNIQUE,
 airbnb(# VerificationDate DATE NOT NULL
 airbnb(# );
[CREATE TABLE
[airbnb=# 
```

WishList

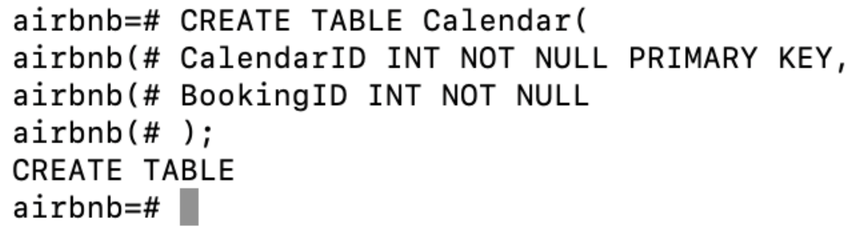


```
[airbnb=# CREATE TABLE WishList(
 airbnb(# WishListID INT NOT NULL PRIMARY KEY,
 airbnb(# GuestID INT NOT NULL,
 airbnb(# ListingName VARCHAR(200) NOT NULL,
 airbnb(# CreationDate DATE NOT NULL,
 airbnb(# ListingID INT NOT NULL
 airbnb(# );
 CREATE TABLE
 airbnb=# 
```

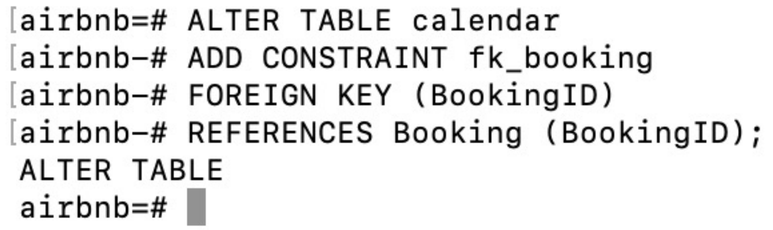


```
[airbnb=# ALTER TABLE wishlist
[airbnb-# ADD CONSTRAINT fk_guest
[airbnb-# FOREIGN KEY (GuestID)
[airbnb-# REFERENCES guestinfo (GuestID);
 ALTER TABLE
[airbnb=# ALTER TABLE wishlist
[airbnb-# ADD CONSTRAINT fk_listing
[airbnb-# FOREIGN KEY (ListingID)
[airbnb-# REFERENCES listing (ListingID);
 ALTER TABLE
 airbnb=# 
```

Calendar

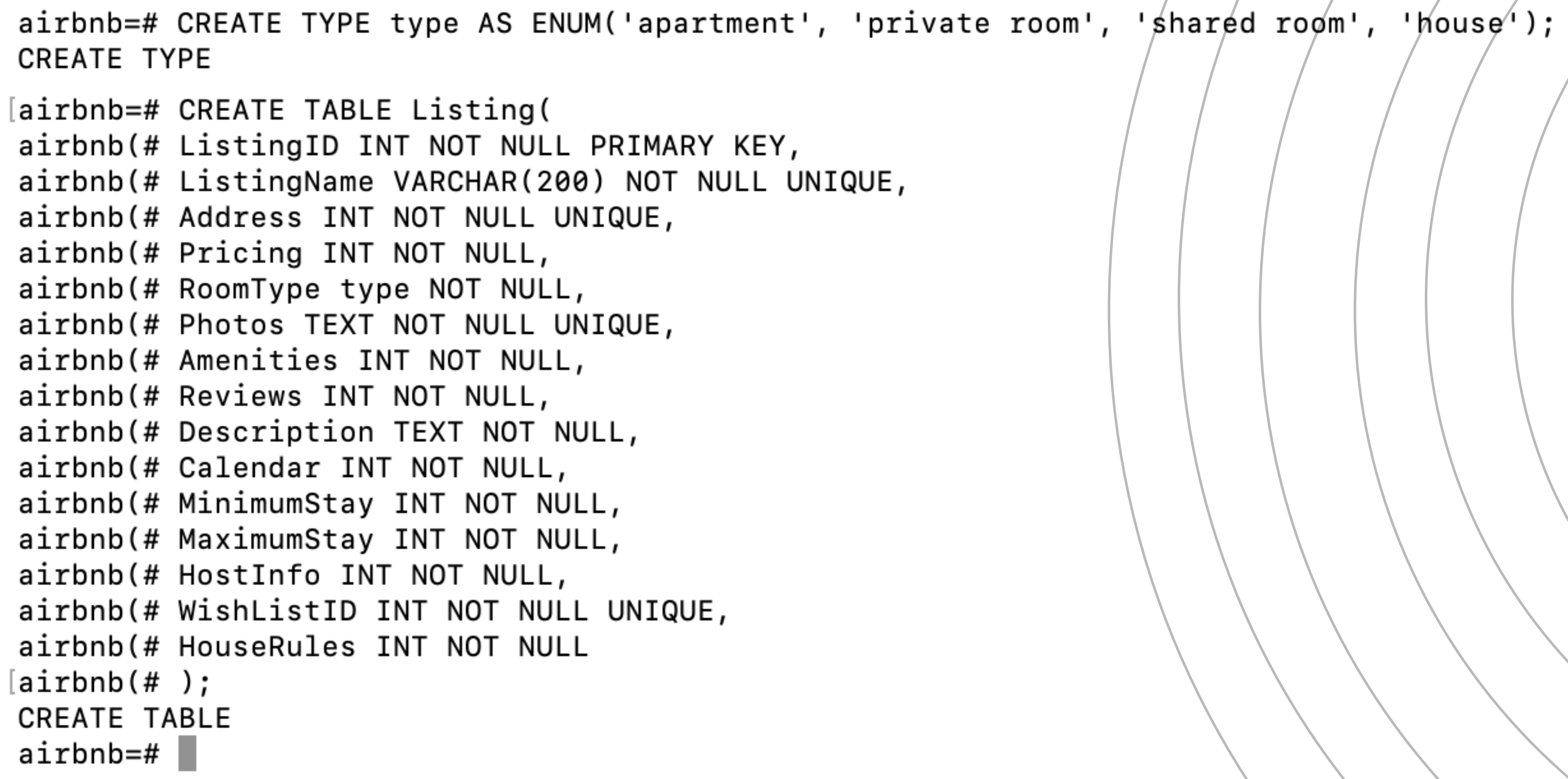


```
airbnb=# CREATE TABLE Calendar(
airbnb(# CalendarID INT NOT NULL PRIMARY KEY,
airbnb(# BookingID INT NOT NULL
airbnb(# );
CREATE TABLE
airbnb=#
```



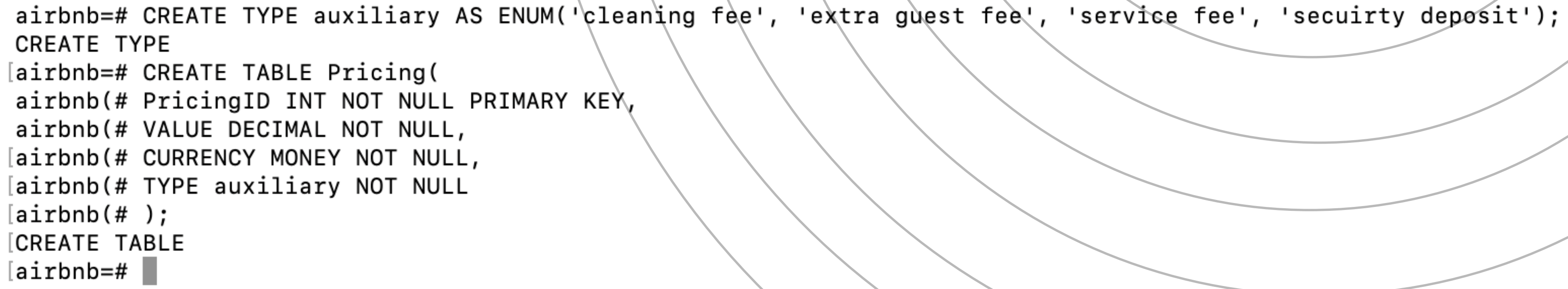
```
[airbnb=# ALTER TABLE calendar
[airbnb-# ADD CONSTRAINT fk_booking
[airbnb-# FOREIGN KEY (BookingID)
[airbnb-# REFERENCES Booking (BookingID);
 ALTER TABLE
 airbnb=#
```

Listing



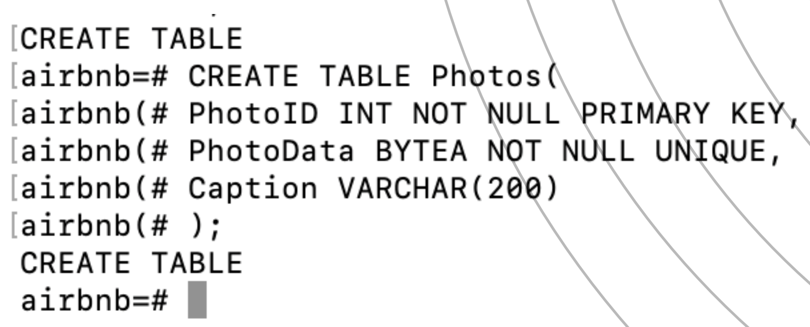
```
airbnb=# CREATE TYPE type AS ENUM('apartment', 'private room', 'shared room', 'house');
CREATE TYPE
airbnb=# CREATE TABLE Listing(
airbnb(# ListingID INT NOT NULL PRIMARY KEY,
airbnb(# ListingName VARCHAR(200) NOT NULL UNIQUE,
airbnb(# Address INT NOT NULL UNIQUE,
airbnb(# Pricing INT NOT NULL,
airbnb(# RoomType type NOT NULL,
airbnb(# Photos TEXT NOT NULL UNIQUE,
airbnb(# Amenities INT NOT NULL,
airbnb(# Reviews INT NOT NULL,
airbnb(# Description TEXT NOT NULL,
airbnb(# Calendar INT NOT NULL,
airbnb(# MinimumStay INT NOT NULL,
airbnb(# MaximumStay INT NOT NULL,
airbnb(# HostInfo INT NOT NULL,
airbnb(# WishListID INT NOT NULL UNIQUE,
airbnb(# HouseRules INT NOT NULL
airbnb(# );
CREATE TABLE
airbnb=#
```

Pricing



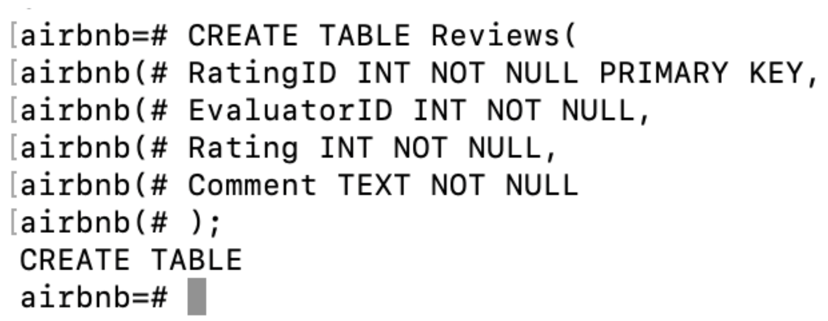
```
 airbnb=# CREATE TYPE auxiliary AS ENUM('cleaning fee', 'extra guest fee', 'service fee', 'secuirty deposit');
 CREATE TYPE
[airbnb=# CREATE TABLE Pricing(
 airbnb(# PricingID INT NOT NULL PRIMARY KEY,
 airbnb(# VALUE DECIMAL NOT NULL,
[airbnb(# CURRENCY MONEY NOT NULL,
[airbnb(# TYPE auxiliary NOT NULL
[airbnb(# );
[CREATE TABLE
[airbnb=#
```

Photos



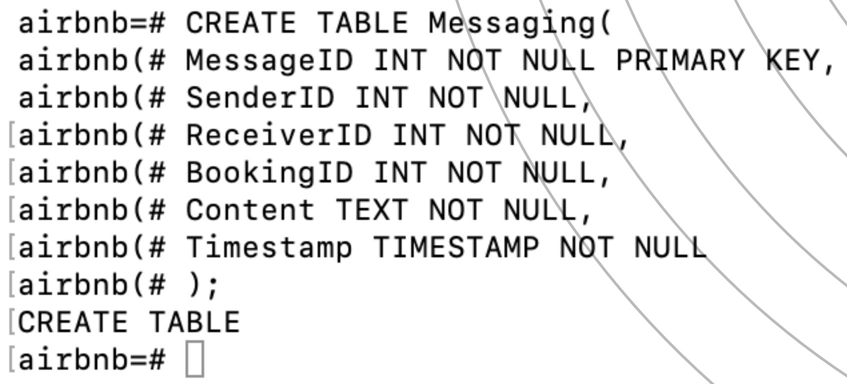
```
[CREATE TABLE
[airbnb=# CREATE TABLE Photos(
[airbnb(# PhotoID INT NOT NULL PRIMARY KEY,
[airbnb(# PhotoData BYTEA NOT NULL UNIQUE,
[airbnb(# Caption VARCHAR(200)
[airbnb(# );
 CREATE TABLE
 airbnb=#
```

Reviews



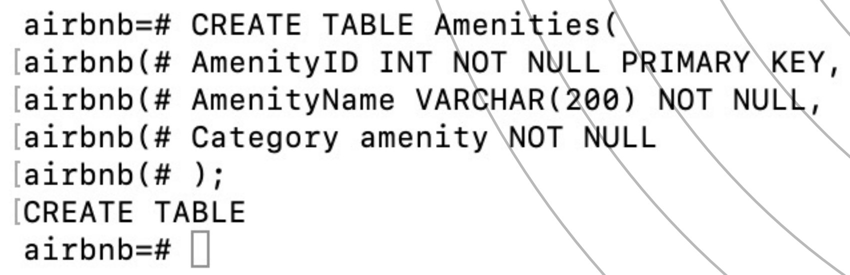
```
[airbnb=# CREATE TABLE Reviews(
[airbnb(# RatingID INT NOT NULL PRIMARY KEY,
[airbnb(# EvaluatorID INT NOT NULL,
[airbnb(# Rating INT NOT NULL,
[airbnb(# Comment TEXT NOT NULL
[airbnb(# );
 CREATE TABLE
 airbnb=#
```

Messaging



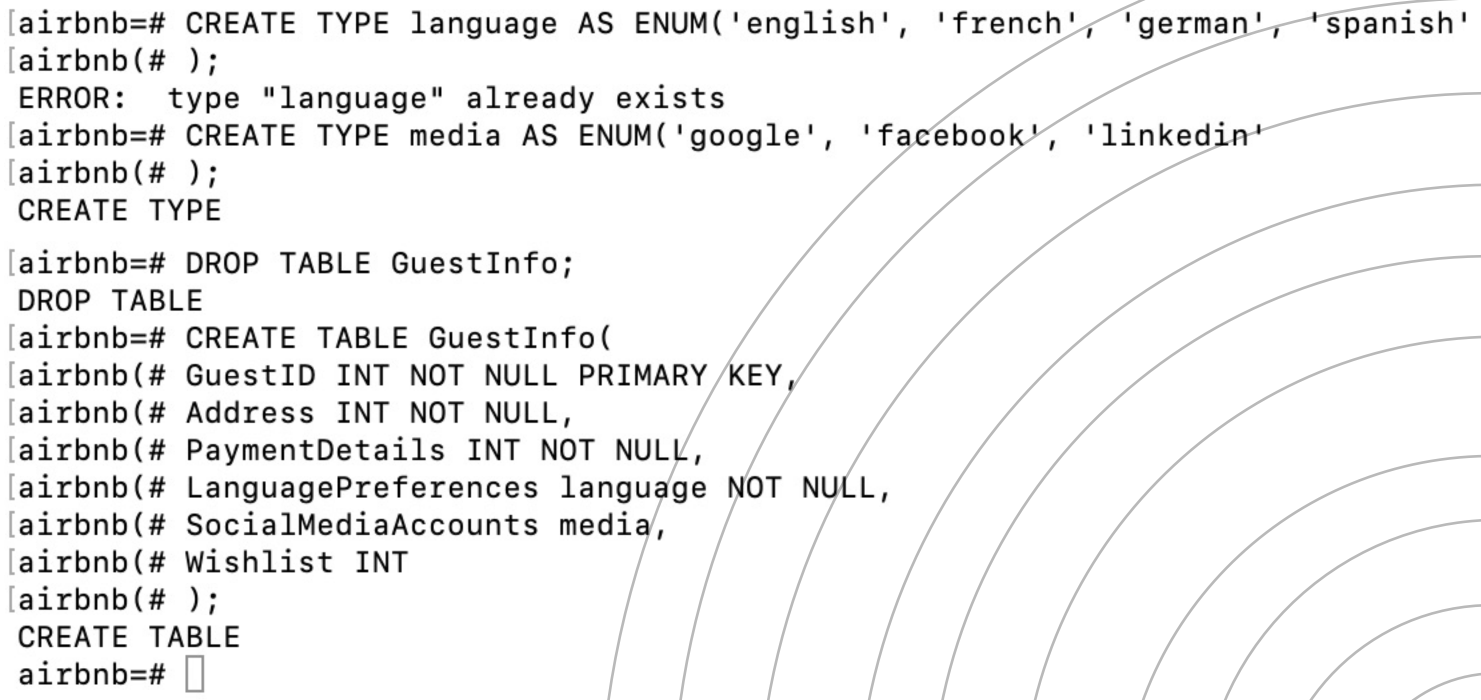
```
 airbnb=# CREATE TABLE Messaging(
 airbnb(# MessageID INT NOT NULL PRIMARY KEY,
 airbnb(# SenderID INT NOT NULL,
[airbnb(# ReceiverID INT NOT NULL,
[airbnb(# BookingID INT NOT NULL,
[airbnb(# Content TEXT NOT NULL,
[airbnb(# Timestamp TIMESTAMP NOT NULL
[airbnb(# );
[CREATE TABLE
[airbnb=# 
```

Amenities



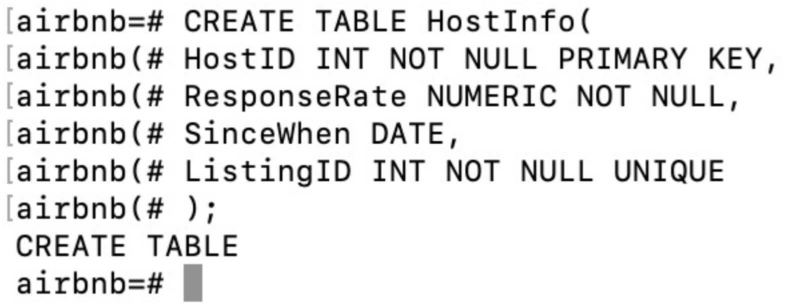
```
 airbnb=# CREATE TABLE Amenities(
[airbnb(# AmenityID INT NOT NULL PRIMARY KEY,
[airbnb(# AmenityName VARCHAR(200) NOT NULL,
[airbnb(# Category amenity NOT NULL
[airbnb(# );
[CREATE TABLE
 airbnb=# 
```

Guest Info

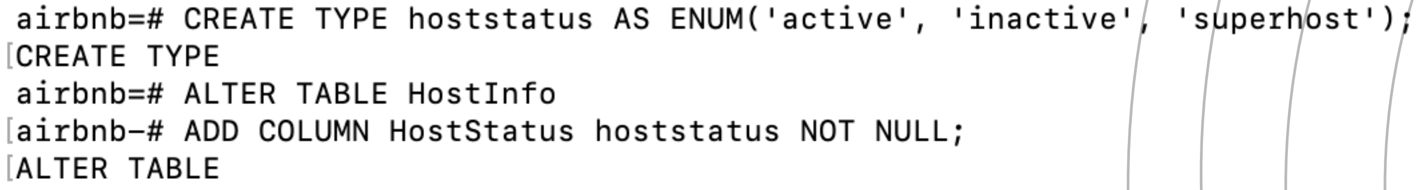


```
[airbnb=# CREATE TYPE language AS ENUM('english', 'french', 'german', 'spanish'
[airbnb(# );
ERROR:  type "language" already exists
[airbnb=# CREATE TYPE media AS ENUM('google', 'facebook', 'linkedin'
[airbnb(# );
CREATE TYPE
[airbnb=# DROP TABLE GuestInfo;
DROP TABLE
[airbnb=# CREATE TABLE GuestInfo(
[airbnb(# GuestID INT NOT NULL PRIMARY KEY,
[airbnb(# Address INT NOT NULL,
[airbnb(# PaymentDetails INT NOT NULL,
[airbnb(# LanguagePreferences language NOT NULL,
[airbnb(# SocialMediaAccounts media,
[airbnb(# Wishlist INT
[airbnb(# );
CREATE TABLE
airbnb=# 
```

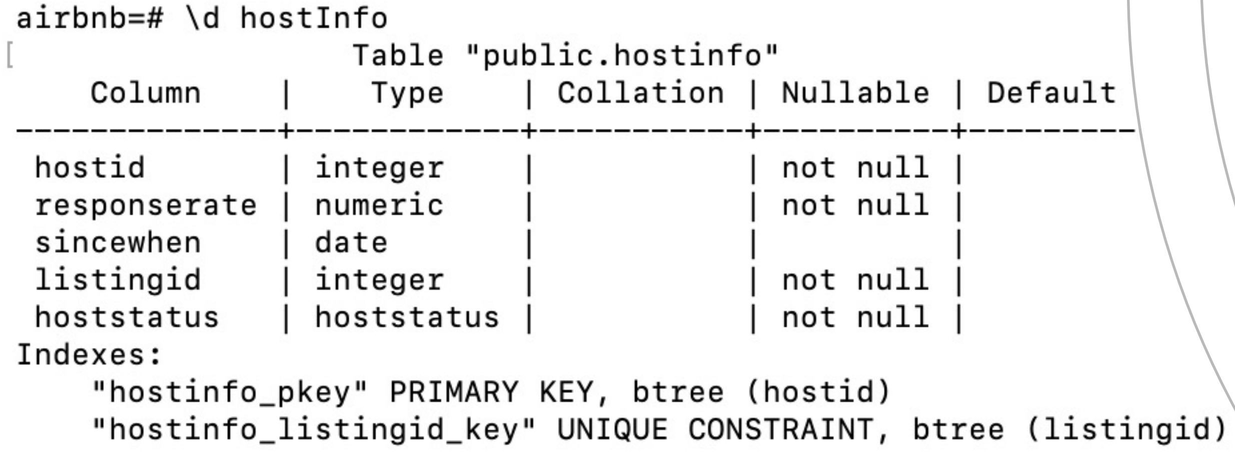
Host Info



```
[airbnb=# CREATE TABLE HostInfo(
[airbnb(# HostID INT NOT NULL PRIMARY KEY,
[airbnb(# ResponseRate NUMERIC NOT NULL,
[airbnb(# SinceWhen DATE,
[airbnb(# ListingID INT NOT NULL UNIQUE
[airbnb(# );
 CREATE TABLE
 airbnb=# 
```

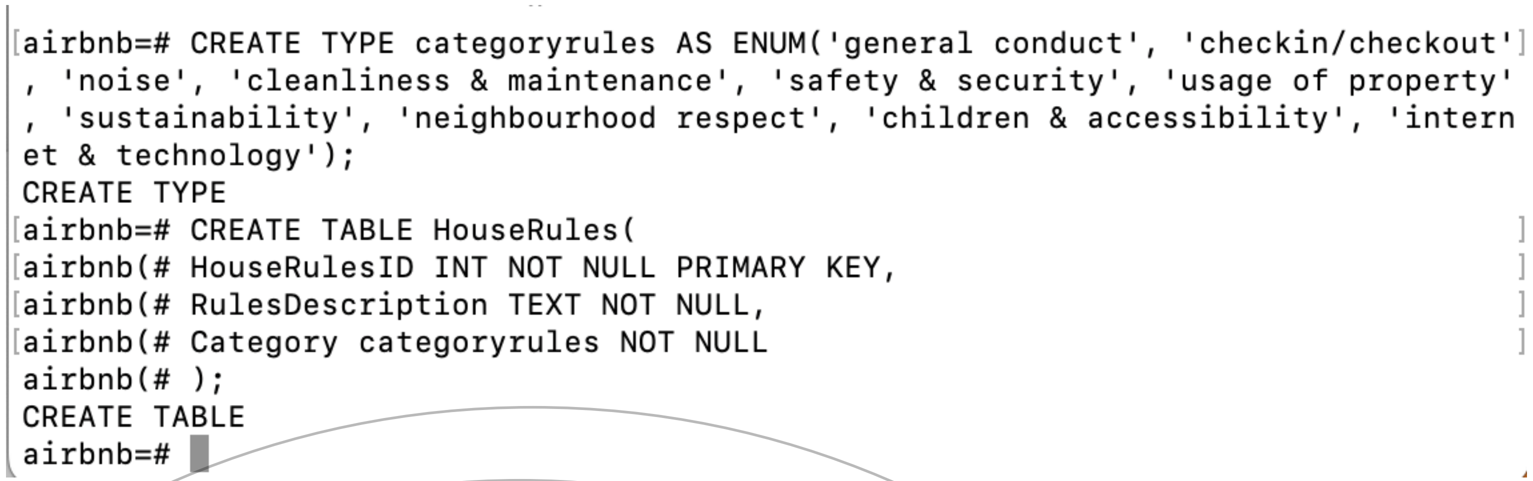


```
 airbnb=# CREATE TYPE hoststatus AS ENUM('active', 'inactive', 'superhost');
[CREATE TYPE
 airbnb=# ALTER TABLE HostInfo
[airbnb-# ADD COLUMN HostStatus hoststatus NOT NULL;
[ALTER TABLE
```



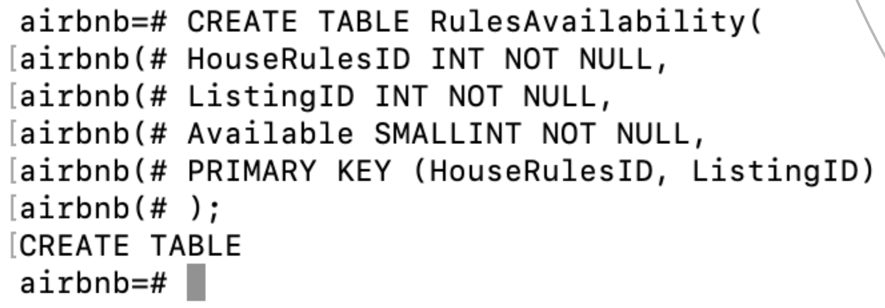
```
 airbnb=# \d hostInfo
[                  Table "public.hostinfo"
    Column    |    Type    | Collation | Nullable | Default
--------------+------------+-----------+----------+---------
 hostid       | integer    |           | not null |
 responserate | numeric    |           | not null |
 sincewhen    | date       |           |          |
 listingid    | integer    |           | not null |
 hoststatus   | hoststatus |           | not null |
Indexes:
    "hostinfo_pkey" PRIMARY KEY, btree (hostid)
    "hostinfo_listingid_key" UNIQUE CONSTRAINT, btree (listingid)
```

House Rules



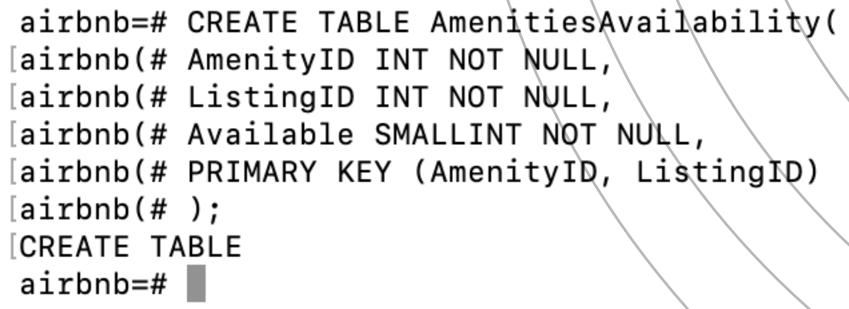
```
airbnb=# CREATE TYPE categoryrules AS ENUM('general conduct', 'checkin/checkout'
, 'noise', 'cleanliness & maintenance', 'safety & security', 'usage of property'
, 'sustainability', 'neighbourhood respect', 'children & accessibility', 'intern
et & technology');
CREATE TYPE
airbnb=# CREATE TABLE HouseRules(
airbnb(# HouseRulesID INT NOT NULL PRIMARY KEY,
airbnb(# RulesDescription TEXT NOT NULL,
airbnb(# Category categoryrules NOT NULL
airbnb(# );
CREATE TABLE
airbnb=#
```

Rules Availability



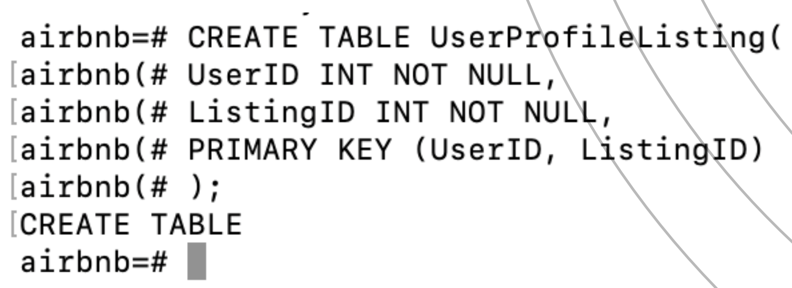
```
 airbnb=# CREATE TABLE RulesAvailability(
[airbnb(# HouseRulesID INT NOT NULL,
[airbnb(# ListingID INT NOT NULL,
[airbnb(# Available SMALLINT NOT NULL,
[airbnb(# PRIMARY KEY (HouseRulesID, ListingID)
[airbnb(# );
[CREATE TABLE
 airbnb=# 
```

Amenities Availability



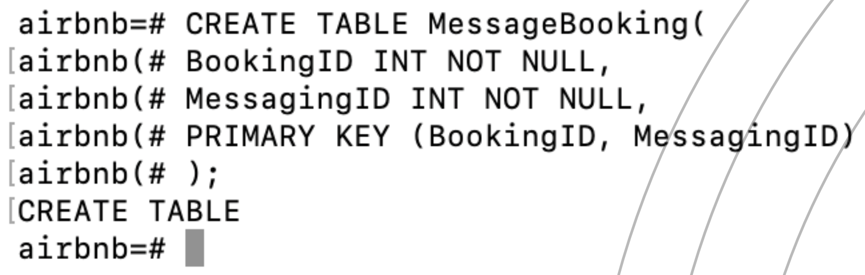
```
 airbnb=# CREATE TABLE AmenitiesAvailability(
[airbnb(# AmenityID INT NOT NULL,
[airbnb(# ListingID INT NOT NULL,
[airbnb(# Available SMALLINT NOT NULL,
[airbnb(# PRIMARY KEY (AmenityID, ListingID)
[airbnb(# );
[CREATE TABLE
 airbnb=# 
```

User Profile Listing



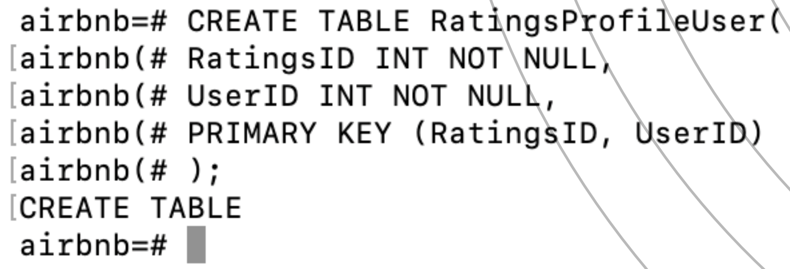
```
 airbnb=# CREATE TABLE UserProfileListing(
[airbnb(# UserID INT NOT NULL,
[airbnb(# ListingID INT NOT NULL,
[airbnb(# PRIMARY KEY (UserID, ListingID)
[airbnb(# );
[CREATE TABLE
 airbnb=# 
```

Messaging Booking



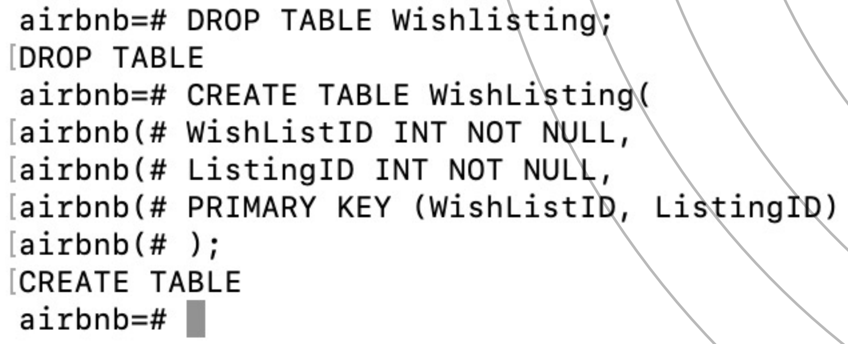
```
 airbnb=# CREATE TABLE MessageBooking(
[airbnb(# BookingID INT NOT NULL,
[airbnb(# MessagingID INT NOT NULL,
[airbnb(# PRIMARY KEY (BookingID, MessagingID)
[airbnb(# );
[CREATE TABLE
 airbnb=# 
```

Ratings Profile User



```
 airbnb=# CREATE TABLE RatingsProfileUser(
[airbnb(# RatingsID INT NOT NULL,
[airbnb(# UserID INT NOT NULL,
[airbnb(# PRIMARY KEY (RatingsID, UserID)
[airbnb(# );
[CREATE TABLE
 airbnb=#
```

WishListing



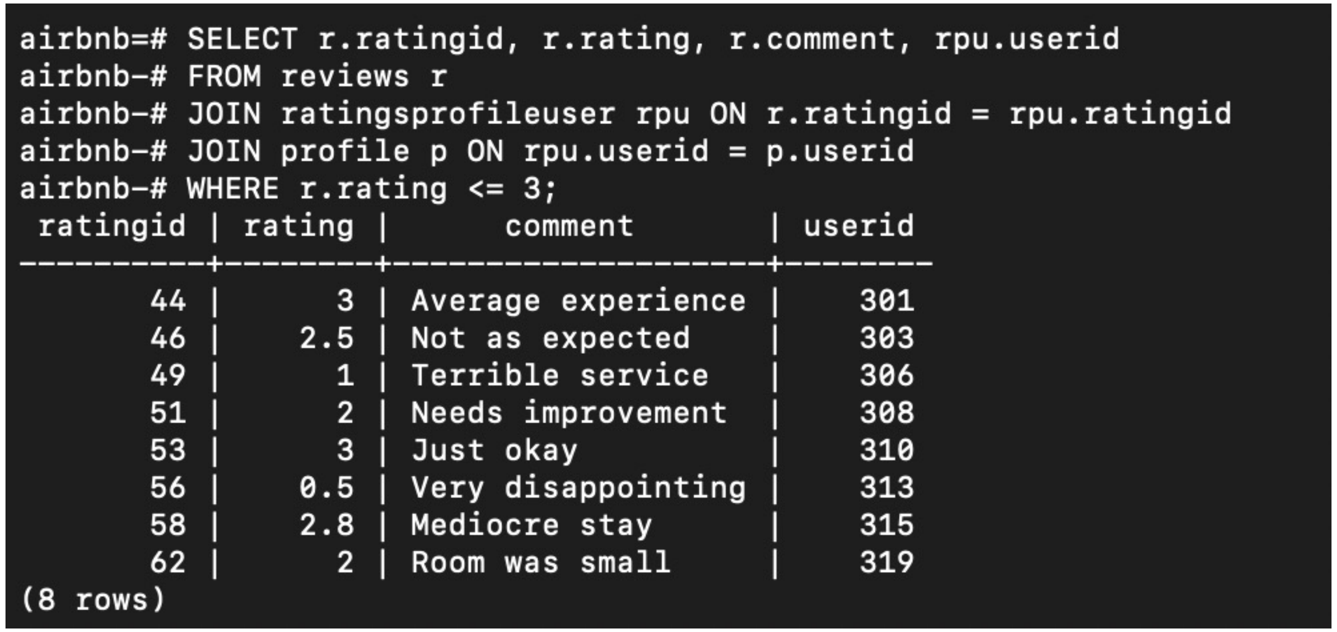
```
airbnb=# DROP TABLE Wishlisting;
DROP TABLE
airbnb=# CREATE TABLE WishListing(
airbnb(# WishListID INT NOT NULL,
airbnb(# ListingID INT NOT NULL,
airbnb(# PRIMARY KEY (WishListID, ListingID)
airbnb(# );
CREATE TABLE
airbnb=#
```

TEST CASES

Ten Test Cases, Mock data generated by Mockaroo

01

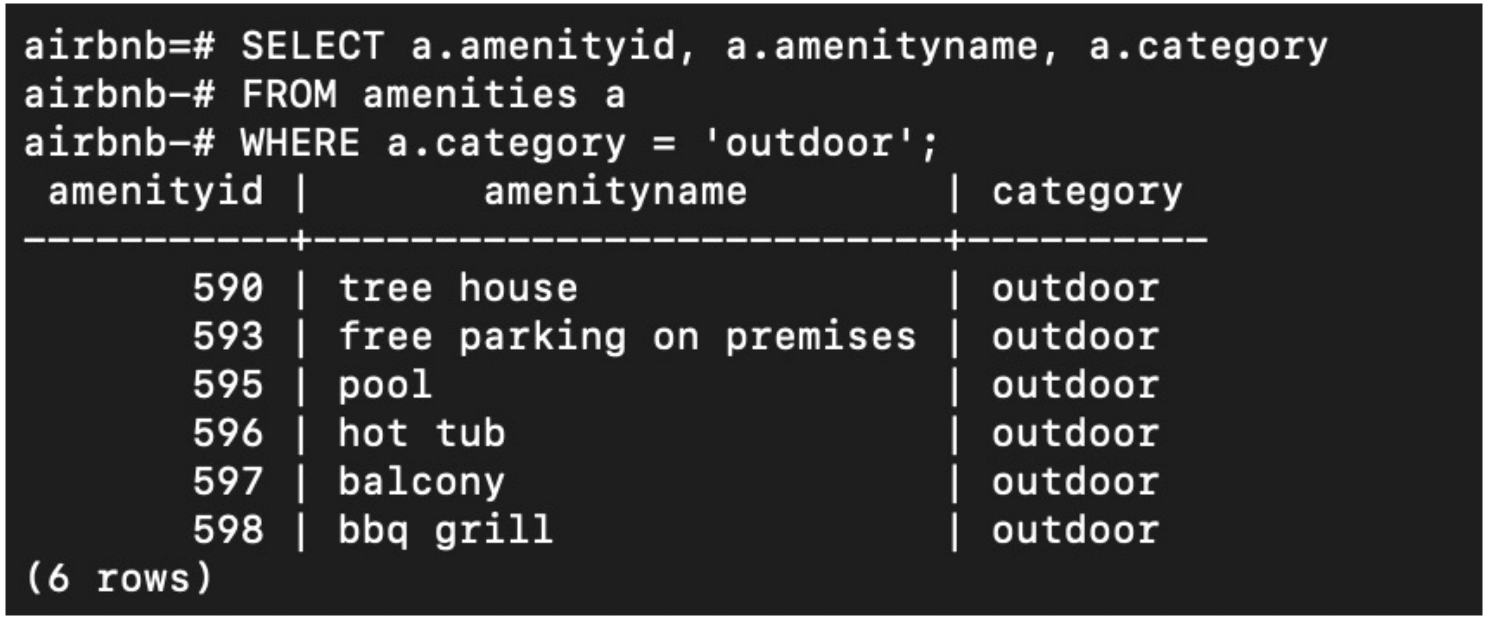
Ratings of 3 and below ☹



```
airbnb=# SELECT r.ratingid, r.rating, r.comment, rpu.userid
airbnb-# FROM reviews r
airbnb-# JOIN ratingsprofileuser rpu ON r.ratingid = rpu.ratingid
airbnb-# JOIN profile p ON rpu.userid = p.userid
airbnb-# WHERE r.rating <= 3;
 ratingid | rating |       comment       | userid
----------+--------+---------------------+--------
       44 |      3 | Average experience  |    301
       46 |    2.5 | Not as expected     |    303
       49 |      1 | Terrible service    |    306
       51 |      2 | Needs improvement   |    308
       53 |      3 | Just okay           |    310
       56 |    0.5 | Very disappointing  |    313
       58 |    2.8 | Mediocre stay       |    315
       62 |      2 | Room was small      |    319
(8 rows)
```

02

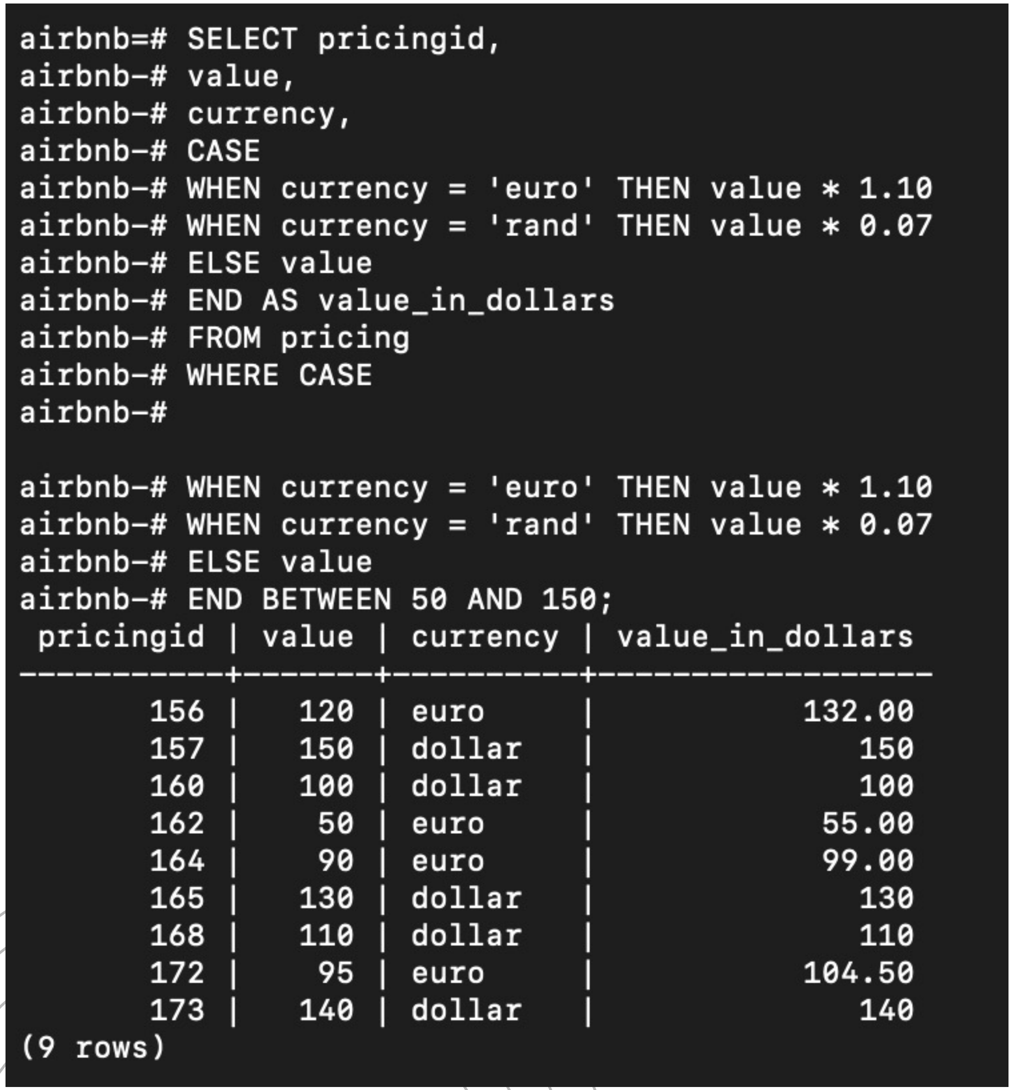
Find listings with outdoor amenities



```
airbnb=# SELECT a.amenityid, a.amenityname, a.category
airbnb-# FROM amenities a
airbnb-# WHERE a.category = 'outdoor';
 amenityid |        amenityname         | category
-----------+----------------------------+----------
       590 | tree house                 | outdoor
       593 | free parking on premises   | outdoor
       595 | pool                       | outdoor
       596 | hot tub                    | outdoor
       597 | balcony                    | outdoor
       598 | bbq grill                  | outdoor
(6 rows)
```

03

Convert all pricing to dollars and find amounts including and in between 50 and 150



```
airbnb=# SELECT pricingid,
airbnb-# value,
airbnb-# currency,
airbnb-# CASE
airbnb-# WHEN currency = 'euro' THEN value * 1.10
airbnb-# WHEN currency = 'rand' THEN value * 0.07
airbnb-# ELSE value
airbnb-# END AS value_in_dollars
airbnb-# FROM pricing
airbnb-# WHERE CASE
airbnb-#

airbnb-# WHEN currency = 'euro' THEN value * 1.10
airbnb-# WHEN currency = 'rand' THEN value * 0.07
airbnb-# ELSE value
airbnb-# END BETWEEN 50 AND 150;
 pricingid | value | currency | value_in_dollars
-----------+-------+----------+------------------
       156 |   120 | euro     |           132.00
       157 |   150 | dollar   |              150
       160 |   100 | dollar   |              100
       162 |    50 | euro     |            55.00
       164 |    90 | euro     |            99.00
       165 |   130 | dollar   |              130
       168 |   110 | dollar   |              110
       172 |    95 | euro     |           104.50
       173 |   140 | dollar   |              140
(9 rows)
```

Phase Two Summary

For the implementation of the Airbnb database I decided to use PostgreSQL via the psql command-line interface. Due to its reputation of reliability and scalbility, PostgrSQL has been implemented by some of the most iconic technology companies – namely, Apply, Reddit, Spotify, Google, and Microsoft to name a few. As I was familiarizing myself with working in the terminal for the first time, the implementation process was iterative. In order to work in the terminal, I needed to learn some shell scripting, as well as the necessary SQL syntax. The repetitive process of creating tables, adding constraints and updating tables with attribute information enabled me to become fluent in these basic queries. Additionally, as I had originally designed my database with mySQL in mind, throughout amending the database, it was necessary to change data types to the required set for PostgreSQL.

The End